Amendments to the Specification:

Please add the following new paragraph after paragraph 16, which starts with "FIG. 5 illustrates" and was added in an amendment in reply to the action of September 29, 2008:

FIG. 6 illustrates an example application of the disambiguation algorithm

Please add the following new paragraphs after paragraph 37, which begins "Block 208 is followed by block 210":

An example technique for determining the meaning of the document is sensing using word sense disambiguation. Before word sense disambiguation and sensing are applied to a document, the following components are used to prepare for meaning analysis: a Tokenizer, a Syntactic Category (part of Speech) Tagger, a Named Entity Recognition and Regular Pattern Identification Module, and a Term Segmenter.

The Tokenizer is responsible for splitting raw data into individual tokens, and for recognizing and marking sentences. This includes handling specific formatting information represented in the document text (for instance, as might appear in HTML tags), as well as identifying specific types of tokens, such as numbers, punctuation, and words. The Tokenizer maintains specific information about the original text, such as a token's byte offset, while stripping some data out (e.g. unnecessary tags), breaking apart some white-space delimited tokens (e.g. pulling a period at the end of a sentence out into a separate token from the word it is adjacent to), or adding some structure to the input text (e.g. sentence annotations). If spell checking is enabled, tokens that are not identified may be matched to correctly spelled candidates, based on the statistical likelihood of the type of error.

When the input contains multiple tokens that are not clearly separated, additional token segmentation processing occurs. World Wide Web domain names are a common example of this type of input. This token segmentation process uses statistical information in order to correctly interpret the domain name "thesearch.com" as a run-together version of: "the search", as opposed to the less likely interpretation: "these arch."

The Part of Speech Tagger analyzes a series of tokens making up a sentence and assigns a syntactic category tag to each token. The Tagger operates on contextual rules that define possible category sequences. The tokens in the series are initialized with the most probable tags for the token as derived from the token data in an Ontology. The tag given to each token can be changed based on the categories around that token. The part of speech data is used during the disambiguation to bias particular meanings of words. For instance, the word "branches" can be either a noun or a verb, and has different meanings in each case ("branches of a tree" vs. "The conversation branches out to ..."). Knowing its part of speech in a specific context narrows down the meanings that are possible.

The Named Entity Recognition and Regular Pattern Identification Module, is responsible for identifying a series of tokens that should potentially be treated as a unit, and that can be recognized as corresponding to a specific semantic type. This module recognizes email addresses, URLs, phone numbers, and dates as well as embodying heuristics for identifying "named entities" such as personal names, locations, and company names. Each recognized unit is marked as a term, and associated with a certain probability that the series should be treated as a unit. In the case of terms that already exist in the Ontology, this probability comes from the system's previous observations of that term. Reference resolution also comes into play at this stage, making it possible, for example, to correctly interpret references to "Mr. Bush" in a document, when the more ambiguous single token "Bush" is used subsequently.

The Term Segmenter goes through the tokens and maps single tokens or sequences of tokens to the terms represented in the Ontology. Competing terms—terms that overlap on one or more tokens—are each given a probability with respect to their competitors.

For instance, for a token sequence "kicked the bucket", there is some probability that the phrase should be treated as a unit (a multi-token term meaning "to die"; "Grandpa kicked the bucket last year"), and some probability that the phrase should be treated as a series of three individual terms (as in, "The toddler kicked the bucket and all the water poured out"). As in other cases, these relative probabilities are determined by the Term Segmenter based on previous observations of those terms. Once each potential term has been identified and labeled, we have

access to the potential meanings associated with each term, and the individual probabilities of those meanings relative to the term as represented in the Ontology.

When these pre-processing steps have been completed, the resulting text can be viewed as a series of probabilistic sets of meaning sets, where each set of meaning sets corresponds to the individual meanings of a particular term. The job of the word sense disambiguation algorithm is then to look at the context established by the juxtaposition of particular terms and meanings in the input text in order to modify the initial context-free probabilities of the meanings into context-dependent probabilities. The result of the application of the algorithm is that the intended meanings of ambiguous words in context should be assigned the highest probability.

Once the above components have processed the document, word sense disambiguation can be performed. The idea underlying the word sense disambiguation algorithm is to utilize known semantic relationships between concepts, as represented in the Ontology, to increase the probability of a particular sense of a word in context -the more words that exist in the context that are related to a particular sense of a word, the more likely that particular sense should be. This follows from the notion of coherence in text, in that a speaker / writer will tend to use related concepts in a single context, as an idea is elaborated or relationships between entities identified.

The methodology used might be described as activation spreading—each meaning in the document sends a "pulse" to the meanings close by in the document that they are related to or associated with. This pulse is used to increase the probability of those meanings. The size of the pulse is a function of the strength of the relationship between the source concept and the target concept, the "focus" of the source concept - that is, how indicative of related concepts a concept can be considered to be (see below) —a measure of term confidence that reflects how confident the system is in the probabilities associated with the meanings of a given term, and potentially the probabilities of the source and target concepts.

The notion of focus is roughly analogous to the specificity of a concept, in that more specific concepts tend to be strongly related to a small set of things, and is somewhat inversely proportional to frequency, since more frequent concepts are less useful for discriminating

particular contexts. However, focus is not directly based on either of those notions. For instance, "Microsoft" refers to a highly specific concept that nevertheless has quite low focus, because its presence in any given document is not highly indicative of that document being about the company or even the domain of computer technology. On the other hand, a very rare term like "thou" also would have quite low focus—although it is rare, it does not strongly influence the interpretation of the words it appears with.

We consider each of the competing terms, and each of their competing meanings, in parallel—each meaning is allowed to influence the surrounding meanings, proportional to their overall probability. This allows meanings that may have a low a priori probability to nevertheless boost particular senses of words around it, so that the context can push a low probability meaning to the top. Several pulsing cycles, in which all the meanings in the document are allowed to spread "activation" to their related meanings, are applied in order to reach a stable state of disambiguation. At each cycle, meanings are boosted, so that the most likely meanings are reinforced several times and end up with the highest probability.

FIG. 6 illustrates an example application of the disambiguation algorithm. Consider the example of the term "Java" 602. "Java" has three possible meanings: "programming language" 604, "coffee" 606, or "an island in Indonesia" 608. Each of the three meanings of this term is initialized with a certain a priori probability that reflects the context-neutral probability of the term. Let's assume that the "programming language" 604 sense of the term is the most likely. When we look at the term in a context in which words such as "milk" 610 and "break" 612 appear, as shown in Figure 6, we find that only the "coffee" 606 meaning is reinforced. This is because there are no relationships between the meanings of the terms around "Java" in either the "programming language" 604 or "island" 608 senses. Through the reinforcement of the "coffee" 606 meaning, and the lack of reinforcement of the other meanings, we will find that the overall probability of the "coffee" 606 meaning will become greater than the other meanings. This can then be viewed as the most likely disambiguation of the term "Java" 602 in that context.

After the application of the word sense disambiguation algorithm, the context-specific probability of each meaning for each term in the input text will be established. This provides a local, term-level view of the meanings conveyed by the text.

However, we would also like to establish a global view of the meaning of the text—a representation of the most important concepts expressed in the text. This has been termed sensing. To achieve this, the system builds on the results of the word sense disambiguation processing, again using semantic relationships as recorded in the Ontology to drive the identification of relevant concepts.

In this case, the goal is to identify the most prominent concepts in the text. This can be viewed as an analogous problem to the problem of identifying the most prominent meaning of a term, moved up from the term level to the document level. As such, the algorithm makes use of the same notion of reinforcement of meanings that the word sense disambiguation algorithm applies.

Related concepts that co-occur in the input text reinforce one another, becoming evidence for the importance of a concept. Implicitly, the algorithm incorporates the notion that more frequent concepts are more important, because concepts that occur more often will be reinforced more often. But unlike many approaches to automated metatagging, this is based solely on data at the semantic level, rather than at the term level.

In approaching the meaning of longer input texts, certain properties of documents are accommodated. In particular, a document may contain sections that are only tangentially related to the main topics of the document. Consider, for example, an HTML document constructed from frames. The "sidebar" frames normally contribute little to the intended interpretation of the document in the "main" frame. Rather than handling this as a special case, it makes sense to treat this as a generic problem that can occur in documents. This is because it often occurs that an author of a document includes something as an aside, without intending it to contribute to the main point, or because of conventions in certain domains, such as the "Acknowledgements" section of academic papers or the "Author bio" section of some magazine articles.

Such sections can interfere with sensing, reinforcing concepts that contribute little to the overall meaning. Furthermore, a document may contain several main points that are addressed in different sections of the document. The algorithm should identify both concepts that are important overall (i.e. recurrent themes of the document), and concepts that are discussed in depth in one portion of the document, but not reinforced in other sections.

Sensing in this case is therefore based on a view of a document as a series of regions. Each region is identified on the basis of certain heuristics, including formatting information. In general, these regions will be larger than the window considered during the sense disambiguation of any given term in the document. Concepts within a region reinforce one another by "pulsing" across ontological relationships between them (proportional to the probability of the concept, derived from the word sense disambiguation, and the strength of the relationship). The most strongly reinforced concepts are selected as the most representative concepts of the region.

The representative concepts across regions in the document are then calculated by considering only the most representative concepts of each region, and allowing them to reinforce one another. At the end of this cycle, a ranked list of meanings important to the document will be produced.

Finally, the relevance of each region to the overall meaning of the document is evaluated, and regions judged to have little relevance (because they do not contain many instances of concepts judged to be most important) are thrown out, and the representativeness of concepts across only the remaining regions is re-calculated. The effect of this is that the main concepts expressed in the document are judged on the basis of only those regions of the document that seem to carry the most semantic weight with respect to those main concepts.

When the text input is short, in the case of a user query to a search engine, for example, a modified form of this sense processing occurs. The "regions" of text used for longer inputs do not exist and therefore cannot be relied on to support an interpretation. Though there is less evidence to rely on, there is a benefit to having a short input: it is possible to explore all potential meanings of an input separately.

For example, when the word "Turkey" appears in a long document, one interpretation of the word will likely dominate as a result of disambiguation and sensing (say, either the bird meaning, or the country meaning). However, if the entire input text is the word "Turkey", the ambiguity of the input can be preserved, and both interpretations passed on to subsequent processing. The huge number of possible permutations of meaning for longer documents makes this infeasible when they are used as the input.

The process results in a ranked list of meanings most representative of the document.